Capstone Design CONFERENCE 2022
JUNE 6–8, 2022 ▶ DALLAS, TEXAS

Adapting SCRUM Ceremonies in Undergraduate Capstone Projects

Hadar Ziv[1], Stavros Kalafatis[2], Luciano Soares[3] and Rafael Prikladnicki[4]
[1]University of California Irvine, USA
[2]Texas A&M, USA
[3]Insper, Brazil
[4]PUCRS University, Brazil

Keywords: Education, capstone projects, agile methods

Corresponding Author: Hadar Ziv, ziv@ics.uci.edu

Between the four authors, we have many years of experience in teaching capstone courses in undergraduate Engineering programs and bringing student-led projects to fruitful completion. Within those, we have a successful track record with computer and engineering projects, where students develop software and hardware for external sponsors with real-world needs, real requirements, and real users. At the end of these projects, student teams deliver working prototypes to their project sponsors, which are often deployed internally within their organizations, or as webapps or cross platform apps, or mobile apps available on an App Marketplace.

One of the top goals of such capstone projects is to prepare students for their future jobs and careers in Information Technology and related fields and to experience the demands, challenges, and uncertainties of "real world" projects as much as possible. This implies not only technical skills in all aspects of the development lifecycle, including software requirements (use cases, user stories), software architecture and design (UML diagrams, etc.), UI/UX design and usability engineering (wireframes, low and high fidelity mockups, etc.), software implementation and testing, and integration and deployment; but also non-technical skills such as professional communication and professional behavior (with their sponsors and teachers), personal dependability and team-work responsibility (towards their teammates), time-management, and project-management skills and tools (such as JIRA, Trello, Asana, Notion.so, etc.)

During these 20+ years, and still today, more and more organizations are shifting their software-development to the Agile framework and, specifically, to the well-known SCRUM method which was first described by Hirotaka Takeuchi and Ikujiro Nonaka in "the new product development Game"[4]. SCRUM is the most popular and heavily used Agile method in real-world software-development, and it inherently follows the principles and values set forth in the Agile Manifesto. It introduces several new roles in software project management (such as SCRUM Master and Product Owner), iterative and incremental activities, artifacts and deliverables and, most relevant to this paper, five ceremonies that should be followed and exercised regularly by SCRUM teams.

Thus, one of our top goals within this topic of software process and project management, is for student teams to follow Agile values and principles, and SCRUM activities and best practices as much as possible including the five SCRUM ceremonies:

- The Sprint: A sprint is a short, time-boxed period when a SCRUM team works to complete a set amount of work. Sprints are at the very heart of SCRUM.
- Sprint planning: During sprint planning, the entire SCRUM team collaborates and discusses the desired high-priority work for the sprint and defines the sprint goal. The SCRUM master's role is primarily to facilitate the meeting. The product owner describes the product goal and answers questions from the team about execution and acceptance criteria/criteria of satisfaction. The team define which work it can accomplish during the sprint.
- Daily SCRUM: The team meets for 15 minutes every day of the sprint to inspect progress toward the sprint goal. They describe for each other how their own work is going, ask for help when needed, and consider whether they are still on track to meet the sprint goal.

- Sprint review: Sprint reviews focus on the product being developed, specifically on the potentially shippable product increment created during the sprint. The SCRUM team invites stakeholders to discuss what was completed during the sprint. They adapt the product backlog as needed based on this feedback.
- Sprint retrospective: Sprint retrospectives focus on the process. The SCRUM team discusses what went right and areas for improvement in the sprint. They make tangible plans for improvements.

There are, however, important differences between a SCRUM team in the real world, versus undergraduate students on project team in a capstone college course[1]. This in turn leads to the interesting challenge of how best to adopt and adapt the five SCRUM ceremonies to student teams in a capstone project course? The top characteristics of capstone-project students include: 1) They are students taking a college course, they are not professional, experienced engineers; 2) They work on their capstone project part-time while taking other classes; they are not full-time employees; and 3) For many students, this is often the first class in their college career that is project-based, team-based, requires self-motivated and self-organizing teams, and professional behavior towards external stakeholders. The top differences between professional developers on a SCRUM team and students on a capstone project team include: 1) task complexity; 2) dependency complexity; and 3) frequency of SCRUM meetings.

Task complexity: Capstone projects are usually defined so that they can be addressed by a team of ~4 students who are working on this project part-time (effectively ~10hr/person/week. As a result, the amount of progress they can make and thus the complexity of the task undertaken is commensurate. Dependency complexity: Since capstone teams have to be evaluated as isolated units, the dependencies are limited to those between the student members. In an industrial setting the development team will have external dependencies that have to be considered as the SCRUM process proceeds. Frequency of SCRUM meetings: in an ideal setting a SCRUM team meets daily to provide updates and proceed. In the capstone educational setting the class conflicts as well as other study related deliverables may reduce the meeting frequency to every other day.

We present next how we chose to adapt each one of the five SCRUM ceremonies:

- The Sprint: This is when students develop the project, sprints may be shorter than usual SCRUM sprints since the academic semester is too short and there are many uncertainties. A 2-week sprint is usually a good amount of time to complete some tasks and quickly identify problems and act on that.
- Sprint Planning: since capstone projects have learning goals, the sprint planning should clearly take in consideration time for students to acquire the knowledge to complete implementation tasks, another issue is due the fact that the team has not maturity to specify a progress velocity, then academic advisors are fundamental to support students on orienting on how they will get prepared for the tasks and how long it should take.
- Daily SCRUM: due the limited time for the team activities, daily meetings are not common, but the recommendation is for students to have a quick meeting at least every other day. In these quick meetings students must bring their progress for the team, but more importantly show what they don't know to continue the project and identify with their colleagues how to learn about the subject: together or alone.
- Sprint Review: reviews are conducted with sponsors; students should present their progress and next steps.
- Sprint Retrospective: Although the retrospective is something for the team members, students are also being evaluated by an advisor. Advisors should bring discussions to point team and members problems and come up with solutions as necessary.

References

1. Jean-Guy Schneider, Peter W. Eklund, Kevin Lee, Feifei Chen, Andrew Cain, and Mohamed Abdelrazek. 2020. Adopting industry agile practices in large-scale capstone education. In Proceedings of the ACM/IEEE 42nd ICSE-SEET '20. Association for Computing Machinery, New York, NY, USA, 119–129. DOI:https://doi.org/10.1145/3377814.3381715
2. H. Ziv and D. J. Richardson, "Constructing Bayesian-network models of software testing and maintenance uncertainties," *1997 Proceedings International Conference on Software Maintenance*, 1997, pp. 100-109, doi: 10.1109/ICSM.1997.624236.
3. https://www.linkedin.com/pulse/20140719182303-6227721-the-three-laws-of-software-development-ziv-s-law/
4. Takeuchi, Hirotaka, and Ikujiro Nonaka. "The new new product development game." *Harvard business review* 64.1 (1986): 137-146.